Package ‘atakrig’

October 9, 2023

Type Package

Title Area-to-Area Kriging

Version 0.9.8.1

Description Point-scale variogram deconvolution from irregular/regular spatial support according to Goovaerts, P., (2008) <doi:10.1007/s11004-007-9129-1>; ordinary area-to-area (co)Kriging and area-to-point (co)Kriging.

Imports terra, gstat, sf, foreach, doSNOW, snow, FNN, methods, MASS, Rcpp

Suggests rtop

License GPL (>= 2.0)

NeedsCompilation yes

Encoding UTF-8

LinkingTo Rcpp

Author Maogui Hu [aut, cre],
Yanwei Huang [ctb],
Roger Bivand [ctb]

Maintainer Maogui Hu <humg@lreis.ac.cn>

Repository CRAN

Date/Publication 2023-10-09 13:40:11 UTC

R topics documented:

ataCoKriging *Area-to-area, area-to-point coKriging prediciton, cross-validation.*

Description

Area-to-area, area-to-point coKriging prediciton, cross-validation.

Usage

```
ataCoKriging(x, unknownVarId, unknown, ptVgms, nmax = 10, longlat = FALSE,
    oneCondition = FALSE, meanVal = NULL, auxRatioAdj = TRUE,
    showProgress = FALSE, nopar = FALSE, clarkAntiLog = FALSE)

atpCoKriging(x, unknownVarId, unknown0, ptVgms, nmax = 10, longlat = FALSE,
    oneCondition = FALSE, meanVal = NULL, auxRatioAdj = TRUE,
    showProgress = FALSE, nopar = FALSE)

ataCoKriging.cv(x, unknownVarId, nfold = 10, ptVgms, nmax = 10, longlat = FALSE,
    oneCondition = FALSE, meanVal = NULL, auxRatioAdj = TRUE,
    showProgress = FALSE, nopar = FALSE, clarkAntiLog = FALSE)
```

Arguments

`x` discretized areas of all variables, each is a discreteArea object.

`unknownVarId` variable name (charaster) defined in x for prediction.

`unknown` a discreted discreteArea object or data.frame[areaId,ptx,pty,weight] to be predicted.

`unknown0` for points prediction or data.frame[ptx,pty] (one point per row) to be predicted.

`nfold` number of fold for cross-validation. for leave-one-out cross-validation, nfold = nrow(x[[unknownVarId]]$areaValues).

`ptVgms` point-scale direct and cross variograms, ataKrigVgm object.

`nmax` max number of neighborhoods used for interpolation.

`longlat` coordinates are longitude/latitude or not.

`oneCondition` only one contrained condition for all points and all variables, \sum_i=1^n\lambda_i +\sum_j=1^m\beta_j =1, assuming expected means of variables known and constant with the study area.

`meanVal` expected means of variables for oneCondition coKriging, data.frame(varId,value). If missing, simple mean values of areas from x will be used instead.

`auxRatioAdj` for oneCondition Kriging, adjusting the auxiliary variable residue by a ratio between the primary variable mean and auxiliary variable mean.

`showProgress` show progress bar for batch interpolation (multi destination areas).

`nopar` disable parallel process in the function even if ataEnableCluster() has been called, mainly for internal use.

`clarkAntiLog` for log-transformed input data, whether the estimated value should be adjusted(i.e. exponentiation).

Value

estimated value of destination area and its variance.

References

Clark, I., 1998. Geostatistical estimation and the lognormal distribution. Geocongress. Pretoria, RSA., [online] Available from: http://kriging.com/publications/Geocongress1998.pdf. Goovaerts, P., 2008. Kriging and semivariogram deconvolution in the presence of irregular geographical units. Mathematical Geosciences 40 (1): 101-128. Isaaks, E. H., Srivastava, R. M., 1989. An introduction to applied geostatistics. New York, Oxford University Press.

See Also

deconvPointVgmForCoKriging, deconvPointCrossVgm, ataKriging

Examples

```
library(atakrig)
library(terra)

## demo data ----
rpath <- system.file("extdata", package="atakrig")
aod3k <- rast(file.path(rpath, "MOD04_3K_A2017042.tif"))
aod10 <- rast(file.path(rpath, "MOD04_L2_A2017042.tif"))

aod3k.d <- discretizeRaster(aod3k, 1500)
aod10.d <- discretizeRaster(aod10, 1500)
grid.pred <- discretizeRaster(aod3k, 1500, type = "all")

aod3k.d$areaValues$value <- log(aod3k.d$areaValues$value)
aod10.d$areaValues$value <- log(aod10.d$areaValues$value)

## area-to-area Kriging ----
# point-scale variogram from combined AOD-3k and AOD-10
aod.combine <- rbindDiscreteArea(x = aod3k.d, y = aod10.d)
vgm.ok_combine <- deconvPointVgm(aod.combine, model="Exp", ngroup=12, rd=0.75)

# point-scale cross-variogram
aod.list <- list(aod3k=aod3k.d, aod10=aod10.d)
aod.list <- list(aod3k=aod3k.d, aod10=aod10.d)
vgm.ck <- deconvPointVgmForCoKriging(aod.list, model="Exp", ngroup=12, rd=0.75,
                                    fixed.range = 9e4)

# prediction
ataStartCluster(2) # parallel with 2 nodes
pred.ataok <- ataKriging(aod10.d, grid.pred, vgm.ck$aod10, showProgress = TRUE)
pred.ataok_combine <- ataKriging(aod.combine, grid.pred, vgm.ok_combine, showProgress = TRUE)
pred.atack <- ataCoKriging(aod.list, unknownVarId="aod3k", unknown=grid.pred,
                  ptVgms=vgm.ck, oneCondition=TRUE, auxRatioAdj=TRUE, showProgress = TRUE)
ataStopCluster()

# reverse log transform
pred.ataok$pred <- exp(pred.ataok$pred)
pred.ataok$var <- exp(pred.ataok$var)
pred.ataok_combine$pred <- exp(pred.ataok_combine$pred)
pred.ataok_combine$var <- exp(pred.ataok_combine$var)

pred.atack$pred <- exp(pred.atack$pred)
pred.atack$var <- exp(pred.atack$var)

# convert result to raster
pred.ataok.r <- rast(pred.ataok[,2:4])
pred.ataok_combine.r <- rast(pred.ataok_combine[,2:4])
pred.atack.r <- rast(pred.atack[,2:4])

# display
pred <- rast(list(aod3k, pred.ataok_combine.r$pred, pred.ataok.r$pred, pred.atack.r$pred))
names(pred) <- c("aod3k","ok_combine","ataok","atack")
plot(pred)
```

ataKriging *Area-to-area, area-to-point ordinary Kriging prediciton, cross-validation.*

Description

Area-to-area, area-to-point ordinary Kriging prediciton, cross-validation.

Usage

```
ataKriging(x, unknown, ptVgm, nmax = 10, longlat = FALSE,
    showProgress = FALSE, nopar = FALSE, clarkAntiLog = FALSE)
atpKriging(x, unknown0, ptVgm, nmax = 10, longlat=FALSE,
    showProgress = FALSE, nopar = FALSE)
ataKriging.cv(x, nfold = 10, ptVgm, nmax=10, longlat = FALSE,
    showProgress = FALSE, nopar = FALSE, clarkAntiLog = FALSE)
```

Arguments

`x` a discreteArea object: list(areaValues, discretePoints), where areaValues: data.frame(areaId,centx,centy,va discretePoints: data.frame(areaId,ptx,pty,weight)

`unknown` a discreted discreteArea object, or just data.frame(areaId,ptx,pty,weight).

`unknown0` for points prediction, data.frame(ptx,pty), one point per row.

`nfold` number of fold for cross-validation. for leave-one-out cross-validation, nfold = nrow(x$areaValues).

`ptVgm` point scale variogram, ataKrigVgm.

`nmax` max number of neighborhoods used for interpolation.

`longlat` coordinates are longitude/latitude or not.

`showProgress` show progress bar for batch interpolation (multi destination areas).

`nopar` disable parallel process in the function even if ataStartCluster() has been called, mainly for internal use.

`clarkAntiLog` for log-transformed input data, whether the estimated value should be adjusted(i.e. exponentiation).

Value

estimated value of destination area and its variance.

References

Clark, I., 1998. Geostatistical estimation and the lognormal distribution. Geocongress. Pretoria, RSA., [online] Available from: http://kriging.com/publications/Geocongress1998.pdf. Goovaerts, P., 2008. Kriging and semivariogram deconvolution in the presence of irregular geographical units. Mathematical Geosciences 40 (1): 101-128. Isaaks, E. H., Srivastava, R. M., 1989. An introduction to applied geostatistics. New York, Oxford University Press. Skøien, J. O. and G. Blöschl, et al., 2014. rtop: an R package for interpolation of data with a variable spatial support, with an example from river networks. Computers & Geosciences 67: 180-190.

See Also

deconvPointVgm, ataCoKriging

Examples

```
library(atakrig)
library(sf)

## load demo data from rtop package ----
if (!require("rtop", quietly = TRUE)) message("rtop library is required for demo data.")
rpath <- system.file("extdata", package="rtop")
observations <- read_sf(rpath, "observations")
observations$obs <- observations$QSUMMER_OB/observations$AREASQKM

## point-scale variogram ----
```

```
obs.discrete <- discretizePolygon(observations, cellsize=1500, id="ID", value="obs")
pointsv <- deconvPointVgm(obs.discrete, model="Exp", ngroup=12, rd=0.75, fig=TRUE)

## cross validation ----
pred.cv <- ataKriging.cv(obs.discrete, nfold=length(observations), pointsv)
names(pred.cv)[6] <- "obs"

summary(pred.cv[,c("obs","pred","var")])
cor(pred.cv$obs, pred.cv$pred) # Pearson correlation
mean(abs(pred.cv$obs - pred.cv$pred)) # MAE
sqrt(mean((pred.cv$obs - pred.cv$pred)^2)) # RMSE

## prediction ----
predictionLocations <- read_sf(rpath, "predictionLocations")
pred.discrete <- discretizePolygon(predictionLocations, cellsize = 1500, id = "ID")
pred <- ataKriging(obs.discrete, pred.discrete, pointsv$pointVariogram)
```

ataSetNumberOfThreadsForOMP

Set number of threads for OpenMP.

Description

Set number of threads for OpenMP.

Usage

```
ataSetNumberOfThreadsForOMP(num)
```

Arguments

num	An integer number of threads for OpenMP.

Details

The deconvolution of variogram is computation intensive. Some parts of them is coded by Rcpp with OpenMP enabled. By default, the number of threads created by OpenMP is the number of local machine cores. It should be noted that OpenMP is not supported for macOS since R 4.0.0.

See Also

ataStartCluster

`ataStartCluster` *Start/stop cluster parallel calculation.*

Description

Start/stop cluster parallel calculation for time consuming prediction. ataIsClusterEnabled queries if cluster connections have been started by ataStartCluster.

Usage

```
ataStartCluster(spec = min(parallel::detectCores(), 8), ...)
ataStopCluster()
```

Arguments

`spec`	A specification appropriate to the type of cluster. See snow::makeCluster. By default, a maximum number of 8 slaves nodes can be creates on the local machine.
`...`	cluster type and option specifications.

`autofitVgm` *Auto fit variogram for points.*

Description

Auto fit variogram for points.

Usage

```
autofitVgm(x, y = x, ngroup = c(12, 15), rd = seq(0.3, 0.9, by = 0.1),
    model = c("Sph", "Exp", "Gau"), fit.nugget = TRUE, fixed.range = NA,
    longlat = FALSE, fig = FALSE, ...)
```

Arguments

`x, y`	values of areas, data.frame(areaId,centx,centy,value).
`ngroup`	number of bins to average from semivariogram cloud.
`rd`	ratio of max distance between points to be considered for bins.
`model`	variogram model defined in gstat::vgms(), e.g. "Exp", "Sph", "Gau".
`fit.nugget`	fit variogram nugget or not.
`fixed.range`	variogram range fixed or not.
`longlat`	indicator whether coordinates are longitude/latitude.
`fig`	whether to plot fitted variogram.
`...`	additional parameters passed to gstat::vgm().

Value

`model`	fitted variogramModel.
`sserr`	fit error.
`bins`	binned gstatVariogram.

Note

The auto-search strategy was derived from `automap::autofitVariogram()`. The function tries different initial values of vgm to find the best fitted model.

`deconvPointVgm` *Point-scale variogram, cross-variogram deconvolution.*

Description

Point-scale variogram, cross-variogram deconvolution.

Usage

```
deconvPointVgm(x, model = "Exp", maxIter = 100,
    fixed.range = NA, longlat = FALSE, maxSampleNum = 100, fig = TRUE, ...)
deconvPointCrossVgm(x, y, xPointVgm, yPointVgm, model = "Exp",
    maxIter = 100, fixed.range = NA, longlat = FALSE,
    maxSampleNum = 100, fig = TRUE, ...)
deconvPointVgmForCoKriging(x, model = "Exp", maxIter = 100,
    fixed.range = NA, maxSampleNum = 100, fig = TRUE, ...)
```

Arguments

`x, y` for `deconvPointVgm` and `deconvPointCrossVgm`, x is a `discreteArea` object.

for `deconvPointVgmForCoKriging`, x is a list of `discreteArea` objects of all variables.

`xPointVgm, yPointVgm`

point-scale variograms of x and y respectively, gstat variogramModel.

`model` commonly used variogram models supported, "Exp" for exponential model, "Sph" for spherical model, "Gau" for gaussian model.

`maxIter` max iteration number of deconvolution.

`fixed.range` a fixed variogram range for deconvoluted point-scale variogram.

`longlat` indicator whether coordinates are longitude/latitude.

`maxSampleNum` to save memory and to reduce calculation time, for large number of discretized areas, a number (maxSampleNum) of random sample will be used. The samples are collected by system sampling method.

`fig` whether to plot deconvoluted variogram.

`...` additional paramters passed to autofitVgm.

Details

The deconvolution algorithm is implemented according to Pierre Goovaerts, Math. Geosci., 2008, 40: 101-128.

Value

`pointVariogram` deconvoluted point variogram.

`areaVariogram` fitted area variogram from area centroids.

`experientialAreaVariogram`
experiential area variogram from area centroids.

`regularizedAreaVariogram`
regularized area variogram from discretized area points and point variogram.

References

Goovaerts, P., 2008. Kriging and semivariogram deconvolution in the presence of irregular geographical units. Mathematical Geosciences 40 (1): 101-128.

See Also

ataKriging,ataCoKriging

Examples

```
library(atakrig)
library(terra)

rpath <- system.file("extdata", package="atakrig")
aod3k <- rast(file.path(rpath, "MOD04_3K_A2017042.tif"))

aod3k.d <- discretizeRaster(aod3k, 1500)
grid.pred <- discretizeRaster(aod3k, 1500, type = "all")

sv.ok <- deconvPointVgm(aod3k.d, model="Exp", ngroup=12, rd=0.8, fig = FALSE)
#pred.ataok <- ataKriging(aod3k.d, grid.pred, sv.ok, showProgress = FALSE)


library(atakrig)
library(sf)

## load demo data from rtop package
#if (!require("rtop", quietly = TRUE)) message("rtop library is required for demo data.")
rpath <- system.file("extdata", package="rtop")
observations <- read_sf(rpath, "observations")

## point-scale variogram
obs.discrete <- discretizePolygon(observations, cellsize=1500, id="ID", value="obs")
pointsv <- deconvPointVgm(obs.discrete, model="Exp", ngroup=12, rd=0.75, fig=TRUE)
```

discretizePolygon *Discretize spatial polygons to points.*

Description

Discretize spatial polygons to points.

Usage

```
discretizePolygon(x, cellsize, id=NULL, value=NULL, showProgressBar=FALSE)
```

Arguments

`x`	a SpatialPolygonsDataFrame object.
`cellsize`	cell size of discretized grid.
`id`	unique polygon id. if not given, polygons will be numbered from 1 to n accroding the record order.
`value`	polygon value. if not given, NA value will be assigned.
`showProgressBar`	whether show progress.

Value

a discreteArea object: list(areaValues, discretePoints).

`areaValues`	values of areas: data.frame(areaId,centx,centy,value), where areaId is polygon id; centx, centy are centroids of polygons.
`discretePoints`	discretized points of areas: data.frame(areaId,ptx,pty,weight), where ptx, pty are discretized points; by default, weight is equal for all points.

Note

Point weight is normalized for each polygon. Weight need not to be the same for all points of a polygon. They can be assigned according to specific variables, such as population distribution.

See Also

discretizeRaster, ataKriging

discretizeRaster *Discretize raster to points.*

Description

Discretize raster to points.

Usage

```
discretizeRaster(x, cellsize, type = "value", psf = "equal", sigma = 2)
```

Arguments

`x`	a SpatRaster object.
`cellsize`	cell size of discretized grid.
`type`	"value", "nodata", "all": whether only valid pixels, or only NODATA pixles, or all pixels extracted.
`psf`	PSF type, "equal", "gau", or user defined PSF matrix (normalized).
`sigma`	standard deviation for Gaussian PSF.

Value

a discreteArea object: list(areaValues, discretePoints).

`areaValues`	values of areas: data.frame(areaId,centx,centy,value), where areaId is polygon id; centx, centy are centroids of polygons.
`discretePoints`	discretized points of areas: data.frame(areaId,ptx,pty,weight), where ptx, pty are discretized points; by default, weight is equal for all points.

Note

Point weight is normalized for each polygon. Weight need not to be the same for all points of a polygon. They can be assigned according to specific variables, such as population distribution.

See Also

discretizePolygon, ataCoKriging

`extractPointVgm` *Extract point-scale variogram from deconvoluted ataKrigVgm.*

Description

Extract point-scale variogram from deconvoluted ataKrigVgm.

Usage

```
extractPointVgm(g)
```

Arguments

`g`	deconvoluted ataKrigVgm object.

Value

a list of gstat vgm model.

`plotDeconvVgm` *Plot deconvoluted point variogram.*

Description

Plot deconvoluted point variogram.

Usage

```
plotDeconvVgm(v, main = NULL, posx = NULL, posy = NULL, lwd = 2, showRegVgm = FALSE)
```

Arguments

`v`	deconvoluted variogram, `ataKrigVgm`
`main`	title
`posx, posy`	position of legend
`lwd`	line width.
`showRegVgm`	show regularized area-scale variogram line or not.

See Also

deconvPointVgmForCoKriging, deconvPointVgm, deconvPointCrossVgm

`rbindDiscreteArea` *Combine two discrete areas.*

Description

Combine two discrete areas.

Usage

```
rbindDiscreteArea(x, y)
```

Arguments

`x, y`	discretized area, list(areaValues, discretePoints).

Value

discretized area, list(areaValues, discretePoints).

`subsetDiscreteArea` *Select discrete area according to area id.*

Description

Select discrete area according to area id.

Usage

```
subsetDiscreteArea(x, selAreaId, revSel = FALSE)
```

Arguments

`x`	a discreteArea object: list(areaValues, discretePoints).
`selAreaId`	area id to select.
`revSel`	reverse select or not.

Value

a discreteArea object: list(areaValues, discretePoints).

`updateDiscreteAreaValue`

Update value of discreteArea object.

Description

Update value(s) of one or some areas of a discreteArea object.

Usage

```
updateDiscreteAreaValue(x, newval)
```

Arguments

`x` a discreteArea object: list(areaValues, discretePoints), where areaValues: data.frame(areaId,centx,centy,va
discretePoints: data.frame(areaId,ptx,pty,weight)

`newval` new values: a dataframe(areaId, value).

Value

a new discreteArea.

Index